Defining the project architecture involves breaking down the system into modules that handle specific functionalities. Here's a suggested high-level architecture for your firewall, IDS, IPS software with a GUI using C++:

1. **Networking Layer:**
   * *Responsibility:* Handle low-level networking tasks, packet inspection, and filtering.
   * *Modules:*
     + **Packet Inspection Module:** Analyzes incoming and outgoing packets, checking for protocol compliance and anomalies.
     + **Filtering Module:** Implements rules to allow or block traffic based on predefined policies.
2. **Rule Engine:**
   * *Responsibility:* Manages firewall policies and intrusion detection/prevention rules.
   * *Modules:*
     + **Firewall Rules Module:** Allows users to define and manage rules for permitting or blocking traffic.
     + **IDS/IPS Rules Module:** Implements rules for detecting and preventing specific types of network attacks.
3. **Logging and Reporting:**
   * *Responsibility:* Record network activities, security events, and generate reports.
   * *Modules:*
     + **Logging Module:** Captures events, including allowed and denied traffic, and stores them securely.
     + **Reporting Module:** Generates reports summarizing security incidents, network usage, and system health.
4. **User Interface (GUI):**
   * *Responsibility:* Provides a user-friendly interface for configuring and monitoring the firewall, IDS, and IPS.
   * *Modules:*
     + **Dashboard Module:** Displays real-time information about the network status, current threats, and system health.
     + **Rule Management Module:** Allows users to define and modify firewall and IDS/IPS rules.
     + **Log Viewer Module:** Enables users to view and analyze logs of network activities.
5. **User Authentication and Access Control:**
   * *Responsibility:* Ensure secure access to the software with proper user authentication and authorization.
   * *Modules:*
     + **Authentication Module:** Verifies user identity before granting access.
     + **Access Control Module:** Manages user roles and permissions.
6. **Automated Response (IPS):**
   * *Responsibility:* Take automated actions to prevent or mitigate detected threats.
   * *Modules:*
     + **Response Module:** Implements automated responses, such as blocking suspicious IP addresses.
7. **Update and Threat Intelligence:**
   * *Responsibility:* Keep the software up-to-date and enhance detection capabilities with external threat intelligence.
   * *Modules:*
     + **Update Module:** Manages software updates to patch vulnerabilities and add new features.
     + **Threat Intelligence Integration Module:** Incorporates external threat intelligence feeds for better threat detection.
8. **Performance Optimization:**
   * *Responsibility:* Ensure the software operates efficiently without significant impact on system performance.
   * *Modules:*
     + **Optimization Module:** Implements performance improvements and tuning mechanisms.
9. **Cross-Platform Compatibility:**
   * *Responsibility:* Ensure the software can run on both Windows and Linux platforms.
   * *Modules:*
     + **Platform Compatibility Module:** Handles platform-specific differences in networking, GUI, and system calls.

This suggested architecture provides a modular and organized structure for your software, making it easier to develop, maintain, and extend in the future. Keep in mind that the specifics may evolve as you proceed with the implementation and gain a deeper understanding of your project's requirements.